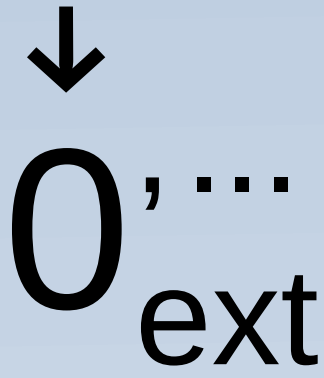
16+

Ivan *Futurologic* Petrov



Hypernullarion: a brief personal representation of the smallest and largest number in the Universe.

The content of this publication is intended for readers fascinated by mathematical abstractions and philosophical reflections.
Recommended for ages 16 and above.

2024

The material presented in this publication is the result of the author's creative effort and reflects their personal perspective. The author developed the described ideas independently and does not claim complete originality. This publication does not aim to serve scientific or educational purposes but represents a private analytical work. The author respects the contributions of other creators and does not seek to challenge the primacy of their ideas. At the time of publication, the author is unaware of any similar works describing these concepts in the same way. The content of the publication has been shaped solely through the author's creative and intellectual activities, relying exclusively on their personal knowledge in this field. Therefore, the author independently created this work, including the description and presentation of the idea. The author is committed to respecting the intellectual property of others and, in the event of discovering similar materials previously published or registered by other authors, acknowledges that all rights and priorities remain with them.

The author of this publication does not encourage any actions and does not intend to offend the feelings of readers. The purpose of this material is to express a personal opinion without aiming to cause misunderstandings or negative emotions. Any resemblance to real persons, names, concepts, terms, events, titles, trademarks, brands, or similar-sounding words from other languages and publications is purely coincidental. The author is not responsible for any potential misunderstandings

The author of this material assumes no responsibility for possible typographical errors, inaccuracies, mistakes, or misinterpretation of the content. All provided information is presented "as is," and the reader is solely responsible for evaluating and using the presented data.

The author of this publication is not responsible for any potential consequences arising from the use of the electronic file, familiarity with the material, or its practical application. Responsibility for interpretation, use, and any resulting outcomes lies solely with the reader. The author also assumes no liability to the reader or third parties for actions based on the content of this material.

The author’s text in this publication has been refined with the assistance of artificial intelligence. The role of artificial intelligence is limited to an auxiliary function—checking and correcting punctuation, stylistic, and grammatical errors. ChatGPT (GPT-3, a large-scale language generation model by OpenAI) was partially used to review the text and enhance the writing style of this editorial article. The author reviewed all edits, adjusted the suggestions provided by ChatGPT at their discretion, verified them, and assumes full responsibility for the content of this publication.

Mathematics can sometimes surprise us with its mysterious ability to create numbers that defy intuition. One such example is the *hypernullarion* — a number that we can call one of the smallest quantities in the world. However, this 'smallest' number is actually much more complex than it may appear at first glance.

What is hypernullarion?

Hypernullarion (denoted as $\overset{\downarrow}{0}_{\text{ext}}^{,\cdots}$) is an ultra-small number obtained using a combination of powerful mathematical concepts such as the Ackermann function, the Busy Beaver theory, and hyperoperations. It tends to zero, but at such a speed that it cannot be expressed in simple words or numbers.

How do we obtain hypernullarion?

To understand how *hypernullarion* arises, we need to delve into several mathematical mechanisms:

1. **Ackermann function** — a function that grows at an astonishing rate and is used to create numbers that far exceed any standard ones. It takes two integers and produces such massive numbers that they become difficult to represent in calculations.
2. **Busy Beaver** — a concept from computability theory that refers to the greatest number of steps a Turing machine can take before halting. It leads to values that cannot be expressed with conventional numbers.
3. **Hyperoperations** — a series of operations starting from basic multiplication and going up to tetration and beyond. In the case of hypernullarion, we apply such operations where the values grow at an incredible speed.

Formula of Hypernullarion

Hypernullarion can be represented by a recursive formula:

$$\overset{\downarrow}{0}_{\text{ext}}^{,\cdots} = \frac{1}{A(M,P(M-1))\uparrow^{k} P(M-1)}$$

where:

- $A(M,P(M-1))$ — is the Ackermann function applied to the values M and $P(M-1)$,
- $\uparrow^{k}$ — is the hyperoperation that defines the ultra-fast growth of values,

- $k=\Sigma(P(M-1))$ — this is the Busy Beaver function, which calculates the maximum number of steps executed by a Turing machine in $P(M-1)$ steps,
- $P(M-1)$ — this is a recursively defined function, dependent on previous values in the sequence,
- $M=\Sigma(P(10))$.

The number $\overset{\downarrow}{0}_{ext}^{,\ldots}$ is defined as the limit of a *super-decreasing* sequence, based on the function $P(n)$, which combines the Ackermann function, Busy Beaver, and hyperoperations.

Thus, $\overset{\downarrow}{0}_{ext}^{,\ldots}$ is a number that approaches zero so quickly that its exact value is virtually impossible to compute, yet it remains strictly defined due to its recursive structure.

Since $\overset{\downarrow}{0}_{ext}^{,\ldots}$ is one of the ultra-small numbers, then $\frac{1}{\overset{\downarrow}{0}_{ext}^{,\ldots}}$ is one of the ultra-large numbers, equal to:

$$\frac{1}{\overset{\downarrow}{0}_{ext}^{,\ldots}}=A(M,P(M-1))\uparrow^{k}P(M-1)$$

Definition of the function $P(n)$

The function $P(n)$ is built recursively:

$$P(n)=\begin{cases}1,\text{if } n=0;\\ A(n,P(n-1))\uparrow^{\Sigma(P(n-1))}P(n-1),\text{if } n>0\end{cases}$$

where:

- $A(m,n)$ this is the Ackermann function, which is defined as:

$$A(m,n)=\begin{cases}n+1,\text{if } m=0;\\ A(m-1,1),\text{if } m>0 \text{ и } n=0;\\ A(m-1,A(m,n-1)),\text{if } m>0 \text{ и } n>0\end{cases}$$

- $\Sigma(k)$ = the maximum number of symbols printed by a Turing machine in k steps,
- $\uparrow^{k}$ this is a hyperoperation, which can be defined through a modification:

$$a\uparrow^{k}b=\begin{cases}a^{b},\text{if } k=1;\\ a\uparrow^{k-1}(a\uparrow^{k-1}(\ldots(a\uparrow^{k-1}(b-1))))_{k-1\text{times}},\text{if } k>1\end{cases}$$

Definition of M

To make M specific but incredibly large, we define it as the outcome of computations associated with the Busy Beaver function. Hence:

$$M=\Sigma(P(10))$$

Here we use a specific value for $n=10$. After we compute $P(10)$, we will obtain a value that is the result of the Busy Beaver function for the number $P(10)$. This will be an enormous number that we will use to calculate the *hypernullarion*.

Conclusion

The number $\overset{\downarrow}{0}^{,\ldots}_{ext}$, we have constructed is an example of an extremely small number, likely surpassing all known numerical values in terms of both computational complexity and growth scale. Its uniqueness lies in the combination of several powerful mathematical concepts: the Ackermann function, Busy Beaver, and hyperoperations. Each subsequent value in the sequence $P(n)$ grows exponentially, becoming so immense that it cannot be expressed using standard methods. This opens new horizons for exploring both large and small numbers in mathematics, providing us with a key to understanding the limits of computational capabilities and theoretical constructions in the world of numbers. $\overset{\downarrow}{0}^{,\ldots}_{ext}$ — it is not merely an abstraction but an intriguing mathematical discovery that prompts us to ponder what lies at the boundaries of human imagination and computational power.

Appendix to the Publication.

Explanation 1.

The formula for the hyperoperation $a\uparrow^k b$ is:

$$a\uparrow^k b=\begin{cases} a^b, \text{if } k=1; \\ a\uparrow^{k-1}(a\uparrow^{k-1}(\ldots(a\uparrow^{k-1}(b-1)))_{k-1 \text{ times}}, \text{if } k>1 \end{cases}$$

represents a recursive definition of hyperoperations, where k ndicates the level of the hyperoperation, and a are b are the numbers for which the operation is computed.

Explanation:

1. **When** $k=1$:
    - This is regular exponentiation: a^b.
    - For example, $2\uparrow^1 3=2^3=8$.
2. **When** $k>1$:
    - In this case, we apply recursive computation of the hyperoperation with a reduction in the level k and the operation, which again utilizes the hyperoperation..

Example for $k=2$**:**

Consider, for example, $a=2$ and $b=3$.

1. For $k=2$:

    $2\uparrow^2 3=2\uparrow^1(2\uparrow^1(3-1))$

2. First, compute $3-1=2$, which is multiplication:

    $2\uparrow^2 3=2\uparrow^1(2\uparrow^1 2)$

3. Now, compute $2\uparrow^1 2$ which is simply exponentiation:

    $2\uparrow^1 2=2^2=4$

4. Substitute the value back:

    $2\uparrow^2 3=2\uparrow^1 4=2^4=16$

Thus, $2\uparrow^2 3=16$.

Example for $k=3$:

Now consider $k=3$, which will lead to more complex computations.

1. For $k=3$:

   $$2\uparrow^3 3 = 2\uparrow^2\left(2\uparrow^2\left(2\uparrow^2(3-1)\right)\right)$$

2. First, compute $3-1=2$, which is multiplication:

   $$\left(2\uparrow^2\left(2\uparrow^2(3-1)\right)\right)=2\uparrow^2\left(2\uparrow^2 2\right)$$

3. Теперь вычислим $2\uparrow^2 2$, which is again exponentiation:

   $$2\uparrow^2 2=2\uparrow^1\left(2\uparrow^1 1\right)=2\uparrow^1 2=2^2=4$$

4. Now we return to computing:

   $$\left(2\uparrow^2\left(2\uparrow^2(3-1)\right)\right)=2\uparrow^2 4=2\uparrow^1\left(2\uparrow^1 3\right)$$

5. Now compute $2\uparrow^1 3$, which again involves exponentiation:

   $$2\uparrow^1 3=2^3=8$$

6. Compute the result $2\uparrow^2 3$:

   $$\left(2\uparrow^2\left(2\uparrow^2(3-1)\right)\right)=2\uparrow^1 8=2^8=256$$

7. Substitute the value back:

   $$2\uparrow^3 3=2\uparrow^2 256=2\uparrow^1(2\uparrow^1 255)$$

8. Compute $2\uparrow^1(2\uparrow^1 255)$:

   $$2\uparrow^1(2\uparrow^1 255)=2^{(2^{255})}$$

Thus, $2\uparrow^3 3=2^{\left(2^{255}\right)}$, this means that the number 2 is raised to the power of 255, and then as a result, 2 is raised to the power of the number obtained in the first step, forming what is called a "power ladder".

Conclusion:

When $k>1$, increases, the hyperoperation requires recursive computation, and with each increase in k the values grow exponentially faster. The example with $k=2$ and $k=3$ demonstrates how rapidly the magnitudes increase, and how the complexity of the calculations grows with each step.

Explanation 2.

Description of the Busy Beaver Function (in the context of the proposed model)

The Busy Beaver function ($\Sigma(n)$) measures the maximum number of ones that a Turing machine can print, starting with a blank tape and running for no more than n steps. It is important to note that for each n, there is a finite maximum value that can be printed, but this value grows extremely quickly as n increases. The Busy Beaver function is a super-exponential function.

For a specific number n the function $\Sigma(n)$ gives the maximum number of ones that can be printed by a Turing machine that performs no more than n steps and halts. A universal Turing machine, starting with *a blank tape and an initial state*, is used to compute the Busy Beaver function.

Formal Definition

For a given number n, the Busy Beaver function is defined as:

$$\Sigma(n) = \text{the maximum number of ones printed by a Turing machine within } n \text{ steps on a blank tape.}$$

These computations are based on a universal Turing machine, which starts with a blank tape and halts after performing n steps. During its operation, it can only print ones (or leave blank cells) on the tape.

It is important to note that the Busy Beaver function is uncomputable. This means that there is no algorithm that can precisely and efficiently compute the value of $\Sigma(n)$ for all n, because for large values of n, the computations become extraordinarily complex.

Real values for $\Sigma(n)$

For small values of n the Busy Beaver function $\Sigma(n)$ has the following known values:

1. **For** $n=1$**:**

   The maximum number of ones that can be printed by a Turing machine in one step is:

   $\Sigma(1)=1$

2. **For** $n=2$**:**

   For two steps, up to 4 symbols can be printed (this is implemented by a more complex Turing machine):

$\Sigma(2)=4$

3. **For** $n=3$**:**

   For three steps, up to 6 symbols can be printed:

   $\Sigma(3)=6$

4. **For** $n=4$**:**

   For four steps, the machine can print up to 13 ones:

   $\Sigma(4)=13$

5. **For** $n=5$**:**

   For five steps, the number of ones printed increases even further:

   $\Sigma(5)=4098$

   In this case, the value $\Sigma(5)$ becomes *extraordinarily* large.

6. **For** $n=6$**:**

   For six steps, an incredibly large number of symbols can be printed:

   $\Sigma(6)\approx 10101034$

   Here, $\Sigma(7)$ is such a large number that it exceeds the limits of standard computational capabilities, and it cannot be expressed in conventional decimal notation.

For our example

In our context, we use the **Busy Beaver function** to compute a value M, which will be incredibly large but well-defined. Specifically, M will be the result of function $\Sigma(n)$, where n is a sufficiently large number chosen in such a way that the result is exceptional.

$M=\Sigma(n)$, where n this is the number for which we compute the value $\Sigma(n)$.

For example, if we choose $n=P(10)$, then:

$M=\Sigma(P(10))$ — this value of M will be astronomically large.

Conclusion

The Busy Beaver function is a powerful tool that allows the construction of enormous numbers with super-exponential growth. For our example, $M=\Sigma(n)$, this number will be so large that it cannot be expressed using standard methods of number notation.

Explanation 3.

Example of calculations for P(n)

- For $n=0$:

  $P(0)=1$

  This is the initial value, as we are setting the base for the recursion.

- For $n=1$:

  $P(1)=A(1,P(0))\uparrow^{\Sigma(P(0))}P(0)=A(1,1)\uparrow^{\Sigma(1)}1=3$

  Here:

  $A(1,1)=3$, as by the definition of the Ackermann function $A(m,n)$ for $m=1$ and $n=1$ gives the result 3.

  $\Sigma(1)=1$, according to the Busy Beaver function for $P(0)=1$ gives the result 1.

  Thus:

  $P(1)=3\uparrow^{1}1=3^{1}=3$

- For $n=2$:

  $P(2)=A(2,P(1))\uparrow^{\Sigma(P(1))}P(1)=A(2,3)\uparrow^{\Sigma(3)}3$

  Here:

  $A(2,2)=7$, as by the definition of the Ackermann function $A(2,2)$ gives the result 7.

- $\Sigma(3)=6$, according to the Busy Beaver function for $P(1)=3$ gives the result 6.

  Thus:

  $P(2)=7\uparrow^{6}2$

  Here: $7\uparrow^{6}2$ this is already a hyperoperation, which gives a result that exponentially exceeds ordinary numbers.

- For n=3:

  $P(3)=A(3,P(2))\uparrow^{\Sigma(P(2))}P(2)$

Here:

- $A(3,P(2))$ this is already an extraordinarily complex computation, which will have a result that greatly exceeds standard values.
- $\Sigma(P(2))$ the Busy Beaver function for $P(2)$, which gives a value so enormous that it cannot be expressed in the usual way.

Thus:

$P(3)=A(3,P(2))\uparrow^{\Sigma(P(2))}P(2)$ this number will be so large that it cannot be expressed using standard methods.

Each subsequent value for $P(n)$ will lead to computations that cannot be explicitly written due to the immense growth of the numbers.